UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/729,607	12/05/2003	Rikin S. Patel	200901531-1	2892

22879 7590 06/04/2009

HEWLETT PACKARD COMPANY
P O BOX 272400, 3404 E. HARMONY ROAD
INTELLECTUAL PROPERTY ADMINISTRATION
FORT COLLINS, CO 80527-2400

EXAMINER	
MADAMBA, GLENFORD J	

ART UNIT	PAPER NUMBER
2451	

NOTIFICATION DATE	DELIVERY MODE
06/04/2009	ELECTRONIC

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Notice of the Office communication was sent electronically on above-indicated "Notification Date" to the following e-mail address(es):

JERRY.SHORMA@HP.COM
ipa.mail@hp.com
jessica.l.fusek@hp.com

UNITED STATES PATENT AND TRADEMARK OFFICE

Commissioner for Patents
United States Patent and Trademark Office
P.O. Box 1450
Alexandria, VA 22313-1450
www.uspto.gov

BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

Application Number: 10/729,607
Filing Date: December 5, 2003
Appellant(s): Rikin Patel

Jenni R. Moen (52,038)
For Appellant

EXAMINER'S ANSWER

This is in response to the appeal brief filed February 23, 2009 appealing from the Office action mailed May 16, 2008.

(1) Real Party in Interest

A statement identifying by name the real party in interest is contained in the brief.

(2) Related Appeals and Interferences

The examiner is not aware of any related appeals, interferences, or judicial proceedings which will directly affect or be directly affected by or have a bearing on the Board's decision in the pending appeal.

(3) Status of Claims

The statement of the status of claims contained in the brief is correct.

(4) Status of Amendments After Final

The appellant's statement of the status of amendments after final rejection contained in the brief is correct.

(5) Summary of Claimed Subject Matter

The summary of claimed subject matter contained in the brief is correct.

(6) Grounds of Rejection to be Reviewed on Appeal

The appellant's statement of the grounds of rejection to be reviewed on appeal is correct.

(7) Claims Appendix

The copy of the appealed claims contained in the Appendix to the brief is correct.

(8) Evidence Relied Upon

2005/0015472 A1	CATANIA	01-2005
7,146,544	HSU	12-2006
2004/0039964 A1	RUSSELL	02-2004

(9) Grounds of Rejection

The following ground(s) of rejection are applicable to the appealed claims:

Claims 2, 3-12, and 15-45 are rejected under 35 U.S.C. 103(a) as being unpatentable over Hsu et al (hereinafter Hsu), U.S. Patent US 7,146,544 in view of Catania et al (hereinafter Catania), U.S. Patent Publication US 2005/0015472 A1.

As per Claims 2, 11, 15, 26 and 34, Hsu in view of Catania discloses a method for managing faults in a web service architecture (i.e., web presentation architecture / "WPA") [col 7, L46] comprising:

receiving a service request (e.g., receiving a request for data) [Abstract] in a web service language (i.e., WSDL) [col 6, L54-58], wherein the service request comprises invoking a service over a network [col 3, L59 – col 4, L11] (e.g. "request_48" for

'task/service', such as data query, data entry, data modification, page navigation, etc.) [col 4, L41-45];

translating the service request into a non-web service language (Catania: i.e., Business Processes Execution Language {BPEL}) [Catania: 0068-0070];

executing the service request (e.g., responding to the request) [Abstract] [transmission of a suitable response 150 back to the client 14) [col 4, L41-44] (i.e., shopping cart JavaBean) [col 5, L15-24] [col 5, L65 – col 6, L8];

encountering an exception during the execution, wherein the execution comprises a fault preventing the fulfillment of the service request (i.e., 'exception' or occurrence of errors / error handling) [col 1, L23-40];

persisting the fault (e.g., storing exceptions/errors including error codes in the error catalog_210) [col 8, L36-54]; and

providing a fault response (performing 'error handling' via error handler_208 to handle exceptions, such as determining the 'correct' error message to display) [col 7, L65 – col 8, L5].

While Hsu discloses substantial features of the invention such as the system of claim 1, and in particular a method for managing faults in a web service architecture (i.e., web presentation architecture / "WPA") [col 7, L46], the added feature of the method further comprising translating the service request into a non-web service language is more expressly disclosed by Catania in a related endeavor.

Catania discloses as his invention a system and a method for issuing event notifications to managed objects to receive notification of events, determining whether the event is a fault, determining the source of the fault when the event is a fault, and propagating a status value indicative of the fault to all managed objects tha are affected by the fault [0011-0013]. In particular, Catania discloses the added feature of the method further comprising translating the service request into a non-web service language (i.e., Business Processes Execution Language {BPEL}) [Catania: 0068-0070].

It would thus be obvious to one of ordinary skill in the art at the time of the invention to combine and/or modify Hsu's invention with the added feature of the method further comprising translating the service request into a non-web service language, as disclosed by Catania, for the motivation of providing a system and method that allows a manager to subscribe to selected types of events and notify the at least one manager of the occurrence of the selected types of events [Abstract].

Claims 11, 15, 26 and 34 recite the same limitations as claim 1, are distinguished only by their statutory category, and thus rejected on the same basis.

As per Claims 3 and 16, Hsu discloses a method of claim 2, wherein the service request is received from a service consumer, the service consumer coupled to the network (e.g., client 14) [Fig. 2].

As per Claims 4, 17, 42 and 43, Hsu discloses the method of claim 3, wherein the fault response is provided to a fault service consumer, and wherein the fault service consumer is coupled to the network (e.g., client 14) [Fig. 2].

As per Claims 5, 18 and 44, Hsu discloses the method of claim 4, wherein the fault service consumer is the same as the service consumer (e.g., client 14) [Fig. 2].

As per Claims 6, 21 and 22, Hsu discloses the method of claim 2, wherein persisting the fault comprises labeling the fault with a unique identifier (e.g., error code) [Fig. 4] [col 7, L9-16].

As per Claims 7, 23 and 33, Hsu discloses the method of claim 6, further comprising storing the fault in a database (e.g., Object Cache Manager_114 w/ database) [col 6, L54-65] (e.g, storing exception/error codes in error catalog_210) [col 8, L36-54].

As per Claim 8, Hsu discloses the method of claim 7, further comprising storing multiple faults in the database, the storage comprising storing fault information (e.g, storing exception/error codes in error catalog_210) [col 8, L36-54].

As per Claim 9, Hsu discloses the method of claim 8, wherein providing a fault response comprises providing access to the database, the access operable to permit a user to

track any fault stored in the database (e.g., error catalog 210 may be accessed based on error code) [col 8, L36-54].

As per Claims 10 and 25, Hsu discloses the method of claim 8, wherein providing a fault response further comprises presenting the fault information in a console, the console operable to list the fault information stored in the database [Fig. 4].

As per Claim 12, Hsu discloses the method of claim 11, further comprising a fault service implementation coupled to the fault service interface, the fault service interface operable to retrieve the fault information from the persistent store (e.g., retrieving data for subsequent use in processing later requests) [col 6, L54-67].

As per Claim 19, Hsu discloses the system of claim 12, further comprising a fault network coupled to the network, the fault network operable to couple the service interface, service implementation, persistent store, and fault service interface (WPA_100) [Fig. 2].

As per Claim 20, Hsu discloses the system of claim 11, wherein the persistent store is a database operable to store faults encountered during the performance (e.g., Object Cache Manager_114 w/ database) [col 6, L54-65] (e.g, storing exception/error codes in error catalog_210) [col 8, L36-54].

As per Claim 24, Hsu discloses the system of claim 12, wherein the fault service implementation is further operable to translate the fault information into a web service language (i.e., WSDL) [col 6, L54-58].

As per Claim 27, Hsu in view of Catania discloses the system of claim 26, wherein the web service language is any protocol registered in the Universal Description Discovery and Integration registry.

While Hsu discloses substantial features of the invention such as the system of claim 26, and in particular a method for managing faults in a web service architecture (i.e., web presentation architecture / "WPA") [col 7, L46], added feature of the system wherein the web service language is any protocol registered in the Universal Description Discovery and Integration registry is disclosed by Catania in a related endeavor.

Catania discloses as his invention a system and a method for issuing event notifications to managed objects to receive notification of events, determining whether the event is a fault, determining the source of the fault when the event is a fault, and propagating a status value indicative of the fault to all managed objects tha are affected by the fault [0011-0013]. In particular, Lech discloses the added feature of the system wherein the web service language is any protocol registered in the Universal Description Discovery and Integration registry (i.e., UDDI registry) [Catania: 0009].

It would thus be obvious to one of ordinary skill in the art at the time of the invention to modify the invention of Hsu with the added feature of the system wherein the web service language is any protocol registered in the Universal Description Discovery and Integration registry, as disclosed by Catania, for the motivation of providing a system and method that allows a manager to subscribe to selected types of events and notify the at least one manager of the occurrence of the selected types of events [Abstract].

As per Claim 28, Hsu in view of Catania discloses the system of claim 26, wherein the web service language is a remote procedure call.

While Hsu discloses substantial features of the invention such as the system of claim 26, and in particular a method for managing faults in a web service architecture (i.e., web presentation architecture / "WPA") [col 7, L46], the added feature of the system wherein the web service language is a remote procedure call is disclosed by Catania in a related endeavor.

Catania discloses as his invention a system and a method for issuing event notifications to managed objects to receive notification of events, determining whether the event is a fault, determining the source of the fault when the event is a fault, and propagating a status value indicative of the fault to all managed objects tha are affected by the fault [0011-0013]. In particular, Lech discloses the added feature of the

system wherein the web service language is a remote procedure call (i.e., RPC Handler_124) [Catania: 0055].

It would thus be obvious to one of ordinary skill in the art at the time of the invention to modify the invention of Hsu with the added feature of the system wherein the web service language is a remote procedure call, as disclosed by Catania, for the motivation of providing a system and method that allows a manager to subscribe to selected types of events and notify the at least one manager of the occurrence of the selected types of events [Abstract].

As per Claim 29, Hsu discloses the system of claim 26, wherein the web service language is a HyperText Transfer Protocol (e.g., HTTP) [col 4, L51].

As per Claim 30, Hsu discloses the system of claim 26, wherein the web service language is an application service interface (e.g., web application program) [col 7, L45].

As per Claim 31, Hsu discloses the system of claim 30, wherein the application service interface is Java message service (i.e., JSPs, J2EE) [col 2, L37-54].

As per Claim 32, Hsu discloses the system of claim 26, wherein the web service language is a protocol approved as a web service description language approved by the World Wide Web Consortium (e.g., HTTP) [col 4, L51] (i.e., WSDL) [col 6, L54-58].

As per Claim 35, Hsu discloses the system of claim 26, further comprising a sub-network coupled to the web services module [Fig. 3].

As per Claim 36, Hsu discloses the system of claim 35, further comprising at least one internal system, the at least one internal system coupled to the sub-network and operable to provide information required by the service request [Figs. 2 & 3].

As per Claim 37, Hsu discloses the system of claim 36, wherein the diagnostic module is further operable to identify any faults caused by the at least one internal system (various types of errors / exceptions that may occur) [col 7, L44-65] [col 8, L58 – col 9, L9].

As per Claim 38, Hsu discloses the system of claim 37, wherein the diagnostic module is further operable to communicate any faults to the fault persistence module (error handler/manager_128) [col 7, L9-16].

As per Claim 39, Hsu discloses the system of claim 38, wherein the fault persistence module is further operable to label each fault with a unique identifier (e.g., error code) [Fig. 4] [col 7, L9-16].

As per Claim 40, Hsu discloses the system of claim 39, wherein the fault persistence module is further operable to direct the persistent store to organize each fault by a

unique identifier (e.g., error code) [Fig. 4] [col 7, L9-16].

As per Claim 41, Hsu discloses the system of claim 26, wherein the web service module is further operable to receive a fault status request [col 4, L40-45].

As per Claim 45, Hsu in view of Catania discloses a system for managing faults in a web services architecture comprising:

a system interface operable to receive a service request in a web services format (e.g., WSDL, HTTP) [col 4, L50-51] [col 6, L54-58], the system interface further operable to translate the service request into a non-web service format (Catania: i.e., Business Processes Execution Language {BPEL}) [Catania: 0068-0070];

a service implementation operable to fulfill the service request, generate a fault report, and persist the fault, the persistence comprising storing the fault report in a persistent store (e.g. storing the errors including the error codes in error catalog 210) [Fig. 3] (e.g., 'logging' exceptions) [col 9, L4-9], wherein generating a fault report comprises detecting a fault during the fulfillment of the service request (e.g., identifying, tracking and logging the errors/fault/exception) [col 7, L9-22], and persisting the fault comprises attaching a unique identifier (i.e., error codes) [col 8, L36-54] to the fault report (notification message / report to manager regarding faults and exceptions/errors) [Abstract] ;

a fault service implementation operable to retrieve the fault report from the persistent store and translate the fault report into a web service format (e.g., WSDL); and

a fault service interface operable to receive fault service requests and transmit a fault service response (WPA 100) [Figs. 1 & 2].

While Hsu discloses substantial features of the invention such as the system of claim 1, and in particular a method for managing faults in a web service architecture (i.e., web presentation architecture / "WPA") [col 7, L46], the added feature of the method further comprising translating the service request into a non-web service language is more expressly disclosed by Catania in a related endeavor.

Catania discloses as his invention a system and a method for issuing event notifications to managed objects to receive notification of events, determining whether the event is a fault, determining the source of the fault when the event is a fault, and propagating a status value indicative of the fault to all managed objects that are affected by the fault [0011-0013]. In particular, Catania discloses the added feature of the method further comprising translating the service request into a non-web service language (i.e., Business Processes Execution Language {BPEL}) [Catania: 0068-0070].

It would thus be obvious to one of ordinary skill in the art at the time of the invention to combine and/or modify Hsu's invention with the added feature of the method further comprising translating the service request into a non-web service

language, as disclosed by Catania, for the motivation of providing a system and method that allows a manager to subscribe to selected types of events and notify the at least one manager of the occurrence of the selected types of events [Abstract].

(10) Response to Argument

Claim 45

With regards to the claim, Applicant primarily argues that neither one of the Hsu or Catania prior art references teaches or discloses particular recited claim feature(s) of the claim, which currently recites:

A system for managing faults in a web services architecture comprising:

a system interface operable to receive a service request in a web services format, *the system interface further operable to translate the service request into a non-web service format*;

a service implementation operable to fulfill the service request, generate a fault report, and persist the fault, the persistence comprising storing the fault report in a persistent store, wherein generating a fault report comprises detecting a fault during the fulfillment of the service request, *and persisting the fault comprises attaching a unique identifier to the fault report*;

a fault service implementation operable to retrieve the fault report from the persistent store and *translate the fault report into a web service format*; and a fault service interface operable to receive fault service requests and transmit a fault service response.

With respect to claim 45, Applicant argues that the combination of Hsu and/or Catania does not teach or disclose the particular limitation of a system interface which is further operable to "translate the service request into a non-web service format", as recited by the claim. The Office respectfully disagrees and submits that Applicant has misinterpreted and/or not fully considered all of the teachings of the prior art references applied in the rejection of the claim. The Office also maintains that the above argued limitation is expressly disclosed by the applied prior art references, either individually or in combination, in accordance with the requirements and language of the current claim recitation.

In support of her argument, Applicant remarks that the teachings of Catania "merely disclose that web service requests are sent in a WSDL format". Applicant contends that, in Catania, a requestor must obtain a copy of the WSDL file from a server and then format the request in a proper SOAP format prior to being sent; however, the SOAP format is a web service format. Applicant also remarks that the cited portions of Catania, which expressly discloses a distributed business processes example, in which an auction manager offers a management service that monitors the progress of a

Request for Quotes (RFQ) process 510, and wherein the business process is implemented in the Business Processes Execution Language (BPEL), is not disclosing that the auction manager is operable to receive the service request and then 'translate the service request into a non-web service format' ". Applicant further comments that no 'translation' occurs since auction manager implements the business process in BPEL. The Office respectfully disagrees.

In response to the arguments, the Office firstly notes Applicant's own 'definition' and/or description for what constitutes a "web service" language / format, versus that of a "non-web service" language / format. In this regard, and with reference to Applicant's own background of the claimed invention, Applicant expressly states that:

"…many companies are now, or will soon be, providing services over networks using web service language. ***Web service language*** is generally defined as a "web service description" language (**WSDL**) which may be any uniform computer language for use in conjunction with the Internet or other network approved by the World Wide Web Consortium (W3C) such as ***RPC, HTTP, JMS,*** or other language registered in a Universal Description Discovery and Integration registry. The difficulty in managing faults when web service languages are used generally rests in the fact that *individual system architectures*, and possibly *individual systems within any given architecture*, operate in 'languages' other than web service languages, often a ***proprietary language*** or ***protocol.*** Thus, a system must 'translate' any service request received in a '*web service format*' into the '*actual computer language*' used by the system providing the service.

[Application Background of the Invention: 0003]

Further, with respect to Applicant's own written description for 'defining' the above features, Applicant also makes it clear that:

"…Upon receiving the 'service request', data management system 320 preferably calls on web services module 330, in the case of a request received in a **WSDL**, to 'translate the service request' into the 'language' typically used by the system architecture 302. In any given embodiment, the language employed by system architecture 302 may be a **WSDL**, or ***any other 'computer-readable protocol*** or ***language' used to execute commands within system architecture 302***. Accordingly, web services module 332 is preferably operable to translate any incoming *service requests* (e.g., WSDL) into a ***language*** or ***'protocol'*** necessary to fulfill the requests according to the specifications of system architecture 302.

[Application Written Description: 0027]

Based on the above 'definitions' for a "web service" language / format, it is clear that a "non-web service" language or format for a message is a language or protocol that may be specific, internal and/or proprietary to a user of a particular business service making the request. It is also clear that 'translating' a service message into a "non-web service" language/format ca be interpreted to mean translating from a WSDL language /format to "languages other than *web service* languages" such as a system or business "proprietary language or protocol", or "any other computer-readable language or protocol that can be used to fulfill the request", other than the WSDL language examples defined above (e.g., HTTP, RPC, JMS, etc.).

As such, the Office asserts that at least Catania expressly discloses one such exemplary "non-web service" language / format: *Business Processes Execution Language* (BPEL). As expressly taught by Catania, "BPEL is an XML-based language designed to enable task sharing for a distributed computing environment, even across multiple organizations, using a combination of web services" [Catania: 0070]. In this regard, Catania expressly teaches:

> "Auction manager 500 has an agreement with Companies C1, C2, C3, and C4 in which auction manager 500 'defines' Request for Quote (RFQ) process 510 for Company C1's purchasing service 508 to submit the RFQ, and for Companies C2, C3, and C4 to respond to the RFQ. In one embodiment, RFQ process 510 is *implemented* in the *Business Processes Execution Language* (BPEL)…. A developer formally describes a 'business process' that will take place across the Web in such a way that any cooperating entity can perform one or more steps in the process the same way. In a supply chain process, for example, a BPEL program might describe a ***"business protocol"*** that 'formalizes' the pieces of information in a *product order*, and the *exceptions* that may have to be handled. Other suitable specifications for implementing RFQ process 510 can be utilized, in addition to, or instead of, BPEL."
>
> [Catania: 0070]

The Office also notes with emphasis the 'definition' of BPEL, as formally defined by Webopedia for example:

> ***Business Process Execution Language (BPEL)*** for Web Services is an XML-based language for standardizing business processes in a distributed or grid computing environment that enables separate businesses to interconnect their applications and share their data. Designed as a combination of IBM's WebServices Flow Language and Microsoft's XLANG spec, platform-independent BPEL allows enterprises to keep ***internal business protocols***

'separate' from *cross-enterprise protocols* so that ***internal processes*** can be changed without affecting the exchange of data from enterprise to enterprise. A BPEL document, for example, keeps track of all the business processes that are connected to a transaction and ensures that the processes are executed in the correct order through the automation of messages.

Based on the disclosures by Catania and the formal definition for BPEL above, it is clear that the exemplary embodiment of a client/consumer *Request for Quote* (RFQ) 'business process', implemented in a BPEL programming 'language or business specific protocol' (and used by Companies C1-C4 and/or Auction Manager 500) is a "computer-readable language or protocol" ***other than*** the "web-service" languages defined and described above for Applicant's claimed invention. As expressly taught by Catania, the RFQ business process request/response interaction between auction manager 500 and Companies C1-4 is implemented in BPEL, but "takes place across the web", and operates in concert or in combination with Web services [Catania: 0068 & 0071-0079] [Figs. 8-12]. It is thus clear that any 'Request for Quotes' *messages* originating from or initiated by Purchasing Service 508 (Company C1) [Fig. 8] are necessarily in BPEL 'format' first (or some other 'internal' business-specific programming language / protocol), and then adapted / transformed ('translated') into a proper 'web-based' *request* message (e.g., a message formatted according to a WSDL 'language' such as HTTP, RPC, or SOAP message), which can be transmitted, received, and processed by the web services 'invoked' to fulfill and provide the *response* in the 'appropriate' programming language used by the service consumer/requestor (i.e., naturally 'BPEL', or other business-specific language). Indeed, this is how "web services invocation"

methodology typically operates and at least this much is obvious to one of ordinary skill in the art.

As support for this position, the Office provides supplemental / evidentiary prior art to *Russell* et al – cited but not referred to - which expressly details the well-known feature of converting or 'translating' request / response messages from a 'web services' format to a 'non-web services' format - and vice versa - through 'serialization' / deserialization' of a message object (e.g., 'JavaBean') for application to web services [0004-0006] [0019-0020] [0040-0048] [Figs. 2,5,6 &10]. For example, and with reference to Fig. 10, Russell expressly discloses an exemplary embodiment and methodology wherein a sample run-time environment may optionally be used for efficiently 'integrating' client code (internal or legacy business programming code) with web services, and wherein client 1000 issues a 'request' that is encoded as a JavaBean. The input request is then 'serialized', creating a 'serialized JavaBean encoded in a SOAP message format' (a web services language / format). The *message* is sent through the network 1020 and then reaches the target Web Service 1030. The web service then interprets (translates) and operates on the data encoded in message 1025 (which is an XML Literal Version of a JavaBean), including data values set by the client for its request in message 1005. When the processing to be carried out by the Web service is finished, the Web service may generate a 'response' message formatted as a SOAP XML document. Upon receiving the SOAP XML message, the response message is 'transformed' (i.e,

deserialized) into the proper complex object format for the client (i.e., JavaBean or similar structured object) [Russell: 0086-0092].

In this regard, the Office also notes with emphasis that the 'RFQ business process embodiment' of Catania's invention is likewise disclosed and essentially identical to a 'preferred embodiment' of Applicant's claimed invention, wherein "service consumer 210 may 'request a *price quote*' on a consumer item" [Application Specification: 0019-0021], and wherein the embodiments fundamentally operate in the same manner. In light of the above, the Office asserts that the argued feature of "translating a service request into a *non-web service* format" is thus expressly disclosed by at least Catania, and the Office maintains its rejection of the claims for at least the reasons provided.

Moreover, with respect to Applicant's argument that Catania does not disclose the recited feature of 'translating a service request into a 'non-web service' format, the Office secondly remarks and points out that Applicant's description for the claimed invention expressly states that

> "in one embodiment, the 'service request' is received by the business service interface in a 'web services description language (WSDL), **such as *remote procedure call* (RPC)**, hypertext transfer protocol (HTTP), Java message service (JMS), or any other WSDL registered in a universal description discovery and integration (UDDI) registry…. Upon receiving the 'service request' (i.e., RPC, HTTP, or JMS 'message'), business service interface 240 preferably 'translates' the message, if necessary, from WSDL (e.g., 'RPC') into the

'operating protocol' or 'language' of the service architecture."

[Application Specification 0019-0020]

Based on the above, it is clear that a 'service request' message may be received in a 'web service' language / format such as a 'request message' in RPC format. As such, the Office significantly remarks that Catania likewise expressly teaches and discloses as part of his system that

"RPC Handlers 118 can be included to handle 'messages' containing *responses* and *requests* between manager 102 and managed object 108. *RPC Handler 118* can alleviate the need for a developer to write, transmit, interpret, and correlate messages by hand, and then 'map' them ***to and from*** **'various native programming types' used by the underlying resources.** A common XML messaging protocol for RPCs is Simple Object Access Protocol (SOAP). In some embodiments, a *Java Application Program Interface (API) for XML-based remote procedure calls* (JAX-RPC) can be used to implement RPC Handler 124 and call 'SOAP-based' Web services…"

[Catania: 0056]

Based on the above disclosure, it is clear that Catania expressly discloses RPC Handlers for interpreting and mapping ('translating') a *request* message in RPC format 'to and from' different 'native programming types' or languages, such as the *Java* programming language (which is a native programming language and thus a 'non-web' service language / format). Catania thus expressly teaches and discloses the argued feature of 'translating a service request into a 'non-web service format', and the Office

maintains its rejection of the claim for at least the reasons provided in response to the argument.

Additionally, with respect to the rejection of claim 45, Applicant secondly argues that the combination of the prior art references applied in the rejection of the claims does not teach or disclose a particular feature of the claim, such as a service implementation operable to "persist the fault...wherein persisting the fault comprises attaching a unique identifier to the fault report". The Office respectfully disagrees and submits that Applicant has misinterpreted and/or not fully considered all of the teachings and disclosures of the prior art references used in the rejection of the claims.

In support of his argument, Applicant remarks that although the cited portions of Hsu discloses the use of 'error codes', for example, Hsu only indicates that the error codes are used to identify the type of message to display and further, indicates that the error codes are applied to 'categories of exceptions'. Applicant also remarks that although Hsu discloses that there may be 'three abstract subclasses' of exceptions and that each exception must fall into one of these categories, there is "no indication that either of the error codes or categories are unique to an error instance. The Office respectfully disagrees and maintains that the argued feature is expressly disclosed by at least the combination of Catania and/or Hsu.

In response to the above argument, the Office firstly remarks that the claim recitation requires at most that 'fault' events (exceptions or error conditions) that may occur and be detected as a result of processing a request be 'persisted' ('stored' in a database), and that the storing of the exception or fault further comprises attaching a 'unique identifier' to the fault message or notification. This is clear with respect to Applicant's own description for 'managing fault / error / exception conditions or events', and on which the claim is based [Application Specification: 0013-0014]. Specifically, Applicant expressly states that:

> "If, at step 120, a fault or exception is encountered, at step 130 that fault is 'persisted' through the system. *Fault persistence* may include 'labeling' the fault with a 'unique identifier' such as a fault number ('error number') or letter combination, for persistence through the system."
>
> [Application Specification: 0014]

In this regard, the Office asserts with emphasis that the features of 'detecting a fault, error, or exception', persisting (storing) the fault, and 'reporting the fault' (providing a fault message / notification to a manager of the system is expressly disclosed by at least Catania and/or Hsu, in accordance with the above described embodiment. Catania, for example, expressly teaches as part of his invention that "Event Notification System 100 can inform one or more managers that an 'event' (e.g, fault / error event) has occurred, and that "a notification can also be used to share informational events". Catania also teaches for example that the Event Push Interface of his invention "allows

manager to register to receive a notification when any of a list of *event types* occurs". [Catania: 0024 & 0027]. Significantly, Catania expressly discloses in one embodiment, that "if there is an 'error' in a request or with the processing of a request, a SOAP 'fault message' is returned instead of the response. A SOAP Fault Message' *includes* a *Fault Code*, a *Fault String*, a *Fault Actor*, and an *Error Detail*". Catania also expressly discloses an exemplary 'fault message' reflecting an error in a client request, including the fault 'code' or 'string' uniquely identifying the detected error or exception [Catania: 0048-0051, 0059-0060 & 0066]. The argued feature of a service implementation operable to "persist the fault...wherein persisting the fault comprises attaching a unique identifier to the fault report" is thus expressly disclosed by at least Catania, and the Office accordingly maintains its rejection of the claim for at least the reasons provided above.

As further support for asserting that the argued feature is taught by the combination of Catania and/or Hsu, the Office additionally remarks that Hsu, in a related endeavor, alternatively teaches or discloses the above argued feature. Hsu expressly discloses a system and method for creating applications, such as web applications. The system includes an 'error handler' that performs error handling functionality during the operation of the application is an error occurs during the processing of the request for data based on error data that is stored in an error catalog [Abstract]. Significantly, Hsu likewise expressly teaches and discloses the storing of 'exceptions / errors' in Error Catalog 210, including and further comprising 'attributes' such as an Exception 'Name',

Error Code, and/or Error Info [Hsu: col 7, L9-16] [col 7, L44 – col 8, L5] [col 8, L36-57] Figs. 3, 4 & 5]. The argued feature of a service implementation operable to "persist the fault...wherein persisting the fault comprises attaching a unique identifier to the fault report" is thus also expressly disclosed by at least Hsu in accordance with the claim requirements, and the Office accordingly maintains its rejection of the claim for at least the reasons provided above.

Further, with respect to the rejection of claim 45, Applicant thirdly argues that the combination of the prior art references applied in the rejection of the claims does not teach or disclose a particular feature of the claim, such as a service implementation operable to "translate the fault report into a web service format". The Office respectfully disagrees and submits that Applicant has misinterpreted and/or not fully considered all of the teachings and disclosures of the prior art references used in the rejection of the claims.

In support of his argument, Applicant remarks that although (a) Catania expressly discloses a system for providing a response to web services 'requests', wherein 'messages exchanged' between the requesting client and the web service application comprise 'XML-messages' described using the Web Services Description Language (WSDL), and (b) Hsu expressly discloses reporting faults and providing a fault report, wherein a "error handler or manager 128 functions to track errors, catalog errors in an

error catalog based on error codes, and display error messages; Applicant comments that this is not disclosing of the argued feature of "translating a fault report into a web service format". The Office respectfully disagrees and maintains that the argued feature is expressly maintained by the at least the combination of Catania and Hsu.

In response to the argument, the Office firstly remarks that as noted previously above, Catania expressly discloses that the 'messages exchanged' in web services are "typically XML messages formatted in accordance with SOAP specification", which can be used to invoke web services, and that 'SOAP messages' are one type of 'web services' language / format, in accordance with Applicant's own definition or description for a 'web service' language / format [Catania: 0005-0009]. The Office notes that at least this much is acknowledged by Applicant. Accordingly, it is clear that the request / response 'messages' and/or fault notification / report 'messages' provided to the one or more managers of Catania's or Hsu's invention are necessarily exchanged in a 'web service' language / format (i.e, SOAP, HTTP etc.). Further, it has been previously established by the Office that at least Catania expressly discloses the feature of "translating a request into a non-web service format", and that 'conversion' (translation) of a message from a 'non-web service' to a 'web service' language / format - and vice versa - using 'serialization / deserialization' of data objects is well-known in the art in view of the evidenciary / supplemental prior art to Russell et al. Accordingly, at least Catania or what is well-known in the art expressly teaches the above argued feature of

"translating the fault report into a web service format" and the Office maintains its rejection of the claim for at least these reasons.

Claims 2-5, 11-12, 15-20, 23, 25 and 26

With respect to the claims, Applicant makes the same arguments as in claim 45 above, and remarks that claim 2, for example, is allowable over the Hsu-Catania combination for the same reasons argued for claim 45 -- namely that "Catania merely discloses that web service requests are sent in the WSDL format, and that the request is then formatted in the proper SOAP request format, which is a 'web service' format. Applicant thus argues that the disclosure of using the WSDL registry to format the request in a SOAP format does not discloses the argued feature of "translating the service request into a 'non-web' service language" or format. The Office respectfully disagrees.

In response to the argument, the Office remarks that in response to the argument that the combination of Catania and/or Hsu does not teach or disclose the argued feature, the Office remarks that it has been established previously above for claim 45 that the *firstly* argued feature of "translating the service request into a 'non-web' language or format is, in fact, expressly disclosed by at least Catania. Additionally, the feature of transforming or converting ('translating') a request / response message 'to

and from' a web service format into a non-web service format, and vice-versa, is also "well-known in the art" (in view of evidentiary / supplemental prior art to Russell et al). The rejection of claim 2 and claims 3-5, which are depending on claim 2, are thus accordingly maintained by the Office for at least the same reasons presented for claim 45 above. Independent claims 11 and 26 recite the same limitations as claim 2, and so the independent claims (together with claims 12, 15-20, 23, and 25, depending on claims 11 and 26 respectively) are also thus rejected on the same basis as independent claim 2.

Claims 6, 21-22 and 39-40

With respect to the claims, Applicant argues that the claims depend upon claims 1, 11, and 26, respectively and are allowable over the Hsu-Catania combination for the same reasons as claim 45 above, namely that neither Catania nor Hsu teaches or discloses the feature of "persisting the fault comprises labeling the fault with a unique identifier". The Office respectfully disagrees.

In response to the argument, the Office firstly notes that claim 1 is a 'canceled' claim on the record, and believes that Applicant intended to refer to independent claim 2 instead. As such, the Office rebuts the argued dependent claims with this understanding. In particular, the Office remarks that in response to the argument that the combination of Catania and/or Hsu does not teach or disclose the above feature, the

Office remarks that it has been established previously above for claim 45 that the *secondly* argued feature of "persisting the fault comprises labeling the fault with a unique identifier" is, in fact, expressly disclosed by at least Catania and, alternatively, by Hsu. The rejection of claim 6, which is depending from independent claim 2, is thus accordingly maintained by the Office for at least the same reasons presented in response to the second argued feature of claim 45 above. Claims 21-22 and 39-40 recite the same limitations as claim6, and are also thus rejected on the same basis.

Claim 24

With respect to the claim, Applicant argues that the claim depend upon independent claim 1, and is thus allowable over the Hsu-Catania combination for the same reasons as claim 45 above, namely that neither Catania nor Hsu teaches or discloses the feature of "translating the fault information into a web service format". The Office respectfully disagrees.

In response to the argument, the Office firstly notes that claim 1 is a 'canceled' claim on the record, and believes that Applicant intended to refer to independent claim 2 instead. As such, the Office rebuts the argued dependent claims with this understanding. In this regard, the Office remarks that in response to the argument that the combination of Catania and/or Hsu does not teach or disclose the argued feature, the Office remarks that it has been established previously above for claim 45 that the

thirdly argued feature of "translating the fault information into a 'web service' language or format" is, in fact, expressly disclosed by at least Catania. Additionally, the feature of transforming or converting ('translating') a request / response message 'to and from' a web service format into a non-web service format, and vice-versa, is also "well-known in the art" (in view of evidentiary / supplemental prior art to Russell et al). The rejection of claim 24 is thus accordingly maintained by the Office for at least the same reasons presented for claim 45 above.

Finally, with respect to the rejection of the pending claim set, Applicant argues that the proposed Hsu-Catania combination is improper since one of ordinary skill in the art would not be motivated to make the proposed combination. In support for this argument, Applicant remarks that the combination lacks 'proper motivation'. The Office respectfully disagrees.

In this regard, the Office reminds Applicant that in response to applicant's argument that there is no suggestion to combine the references, the examiner recognizes that obviousness can only be established by combining or modifying the teachings of the prior art to produce the claimed invention where there is some teaching, suggestion, or motivation to do so found either in the references themselves or in the knowledge generally available to one of ordinary skill in the art. See *In re Fine*, 837 F.2d 1071, 5 USPQ2d 1596 (Fed. Cir. 1988)and *In re Jones*, 958 F.2d 347, 21 USPQ2d 1941 (Fed.

Cir. 1992). In this case, Catania and Hsu are at the very least related endeavors which seek to provide a system and/or method of error detection, error classification and storage, and error notification in web services applications. Further, the Office asserts that it is obvious to one of ordinary skill in the art that it would be advantageous to modify the 'fault / error notification' messages provided to the manager of Catania's invention with the additional fault / error 'exception attributes' stored in the Error Catalog of Hsu's invention to uniquely identify faults for storage, retrieval, and management, as well as to provide unique identification and detailed information for each error event detected by the system, as noted in at least Hsu (e.g., error handling). The Office thus maintains that there is proper motivation for combining the references, and the rejection of the claims is maintained for at least this reason.

(11) Related Proceeding(s) Appendix

No decision rendered by a court or the Board is identified by the examiner in the Related Appeals and Interferences section of this examiner's answer.

(12) Conclusion

For the above reasons, it is believed that the rejections should be sustained.

Respectfully submitted,

Glenford Madamba
May 23, 2009

/John Follansbee/
Supervisory Patent Examiner, Art Unit 2451

/Bunjob Jaroenchonwanit/
Supervisory Patent Examiner, Art Unit 2456

Conferees:

John Follansbee
/John Follansbee/

Supervisory Patent Examiner, Art Unit 2451

Jenni R. Moen
Baker Botts LLP
21 Ross Avenue
Suite 600
Dallas, TX 75201-2980